October 2002 Technical Report No. 123

Final Report

On-demand Interactive Simulation-centered Training:

Responsive Technology to Meet Warfighter Training Requirements

Allen Munro
Mark C. Johnson
Quentin A. Pizzini
Josh Walker

Behavioral Technology Laboratories
University of Southern California
250 N. Harbor Drive, Suite 309
Redondo Beach, CA 90277

Developed under funding by:
US ARMY RDECOM, STC

Under Contract No. N61339-02-C-0013

Approved for Public Release: Distribution Unlimited

Reproduction in Whole or in Part is permitted for any purpose of the United States Government

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
30-10-2002	Final Report	23-01-2002 - 30-09-2002

4. TITLE AND SUBTITLE

On-demand Interactive Simulation-centered Training:
Responsive Technology to Meet Warfighter Training Requirements

5a. CONTRACT NUMBER
N61339-02-C-0013

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

6. AUTHOR(S)
Munro, Allen
Johnson, Mark C.
Pizzini, Quentin A.
Walker, Josh

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Southern California
Behavioral Technology Laboratories
250 N. Harbor Dr., Suite 309
Redondo Beach, CA 9077

8. PERFORMING ORGANIZATION REPORT NUMBER
Technical Report No. 123

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

US Army RDECOM, STC
12423 Research Parkway
Orlando, FL 32826-3276

Naval Air Warfare Center, Training Systems Div.
12350 Research Parkway Code 2535
Orlando, FL 32826-3275

10. SPONSOR/MONITOR'S ACRONYM(S)
STRICOM RDECOM-STC NAWCTSD

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release: Distribution is Unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The army needs portable, mobile Embedded Training and simulation technologies to support the future dismounted soldier. The availability of on-demand training will not require hardware features that are not already anticipated for the individual soldier system. In addition to not adding to the soldier's physical burden, effective warfighter training will ordinarily have the following characteristics: task focus, modularity and accessibility, individualization and adaptability, and maintainability. Four objectives were met in this project: (1) Development and demonstration of interactive 2D simulation training using a wearable computer system with head-mounted display; (2) On-demand distribution of 2-D simulation training by delivering over a network in the context of an HTML browser; (3) Demonstration of three distinct modes of interactive instruction in the training environments; and (4) Development and demonstration of interactive behavior authoring features.

15. SUBJECT TERMS
Simulation training, Distance learning, Advanced Distributed Learning, ADL, Authoring, Training architecture, Authored training, Authored simulation, Embedded Training for Warfighters

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Donna Darling
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U		32	(310) 379-0844

Final Report

On-demand Interactive Simulation-centered Training:
Responsive Technology to Meet Warfighter Training Requirements

Contract No. N61339-02-C-0013

Prepared for:

Paul Dumanoir
US ARMY RDECOM, STC
12423 Research Parkway
Orlando, FL 32826

30 October 2002

Allen Munro
Mark C. Johnson
Quentin A. Pizzini
Josh Walker

munro@usc.edu

Technical Report No. 123

Behavioral Technology Lab
Rossier School of Education
University of Southern California
250 North Harbor Drive, Suite 309
Redondo Beach, CA 90277

Table of Contents

List of Figures

List of Tables

1.0 Introduction

The Army of the future will require more skills and the faster acquisition of skills by its personnel. Developing new learning technologies that support the production and delivery of cost-effective training materials when and where needed is essential to the success of future Army missions. Advanced training systems will depend upon new research results in simulation training, cognitive studies, artificial intelligence, natural language processing, and education to assure that effective distance learning can meet the training demands of the Army of tomorrow.

In order for soldiers to operate effectively with the complex army of the future, they will require training. Such training must deal with basic and advanced operations, mode recognition, operations doctrine, field troubleshooting, and many other topics. In some cases, it may be appropriate for soldiers to receive specialized training delivered by advanced soldier systems while they are *en route* to an operation. For example, it may be useful to review the proper operation of certain subsystems during a particular type of mission as part of the preparation for that mission.

This research effort addresses the army's need for portable, mobile Embedded Training (ET) and simulation technologies to support the future dismounted soldier. The availability of on-demand training should require no hardware features that are not already anticipated for the individual soldier system. Advanced training need not cost the dismounted soldier one additional ounce of weight to be carried. In addition to not being a physical burden, effective warfighter training will ordinarily have the following characteristics:

- *Task focus*: The training must be relevant to the soldier's job.
- *Modularity and accessibility*: It must be possible to access just the training required, without wading through irrelevant pedagogical material.
- *Individualization and adaptability*: The training must provide individualized instruction that adapts to provide each soldier with the training that he most requires. In many cases, this requires that each learner get feedback, practice, explanations, and demonstrations that are appropriate to his particular interactions with the training system.
- *Cost-effectiveness*: Unless high technology training can be developed at reasonable cost, very few of the many possible jobs of a soldier will be covered by on-demand embedded training.
- *Maintainability*: Soldier systems evolve. Tactical doctrine can change as a result of advances in sensors or weapons technology. Therefore, training about soldier systems must be modified and maintained to provide updated, truly relevant information.

These characteristics help clarify the features of an ideal set of target training technologies in support of training the dismounted soldier. First, the task focus requirement strongly suggests that a simulation context will often be a necessary component of training. In the context of a behaviorally accurate simulation, a soldier can learn about the possible consequences of different actions in a variety of contexts. This can be accomplished both by having the training system demonstrate how a task is carried out, and by providing a monitored practice environment together with appropriate remediations.

Second, the training must be broken up into small chunks, most of which will not take very much time for most soldiers to work through. This will make it possible for soldiers who have prerequisite knowledge to get directly to the task-oriented training that they need for a particular operation or assignment. There must be a well-structured instructional control system that can determine what pedagogical materials a given student needs to work with, and in what order.

Third, the requirement for individualization and adaptability requires not only that appropriate content objects be selected, but also that there should be individualization of training within content units. A student who demonstrates understanding and appropriate use of knowledge should be able to move on in a lesson, while another soldier who fails to demonstrate such mastery will need to receive remedial instruction or more practice opportunities.

The fourth and fifth requirements, cost-effective development and maintainability, strongly suggest that authoring tools be used in preference to programming whenever feasible. Authoring systems are particularly likely to promote rapid response to changing training requirements and support cost effective training product life cycle extension in the current context of constantly changing software development staffs.

1.1 Research Tasks.

A primary goal of this project has been to show that the iRides development and delivery system, which is based on the newVivids architecture (Munro, Surmon, Johnson, Pizzini, and Walker, 1999), has the potential to meet these requirements for teaching about the assembly and operation of soldier systems. Four specific objectives were established toward that end.

1. Demonstrate interactive 2D simulation training using a wearable computer system with head-mounted display.
2. Demonstrate on-demand distribution of 2-D simulation training by delivering *over a network* in the context of an HTML browser.
3. Demonstrate three distinct modes of interactive instruction in both of the training environments described in objectives 1 and 2.
    - *Introduction* to a task. Explanation and guidance should lead the warfighter through the required steps of the task.
    - *Guided practice*. The soldier should be coached through the correct performance of the task.
    - *Unguided practice* with opportunistic instruction. The soldier should be able to interact with a 2D simulation with behavioral fidelity. When a soldier makes an error that can be used as an opportunity for instruction, that instruction should be presented.
4. Demonstrate interactive behavior authoring features.

The iRides and related research carried out at USC (Munro, 1994; Pizzini, Munro, Wogulis & Towne, 1996; Munro, Johnson, Pizzini, Surmon, Towne & Wogulis, 1997; Pizzini, Munro, Wogulis & Towne 1996; Munro & Pizzini, 1998; Johnson, Rickle, Stiles & Munro, 1998; Munro, 1998, 1999) has as a goal the union of a detailed theory of task-centered instruction with a practical system for developing and delivering instruction. Proof-of-concept training modules are to incorporate instructional contexts such as interactive graphical simulations as an aid to teaching warfighters how to perform complex jobs well. Authoring tools and data-driven delivery systems are to constitute a rigorous embodiment of the theory. The training development tools are to support cost-effective training development and training maintainability.

1.2 Research Products.

Research products from this RDECOM-sponsored project include the following.

1. The iRides delivery infrastructure was enhanced with WebStart. This is an advanced method for efficiently delivering training content over the web
2. A set of simulation-centered training materials on the Phillips Medical Automatic Electronic Defibrillator was developed. These web-delivered training materials and the two following sets of materials can be accessed from **`http://btl.usc.edu/WS/index.html`**.
3. Simulation-centered training materials on the Land Warrior components and their connections were developed and made available at the above web site.
4. A web-accessible simulation of the Land Warrior GPS system was also developed and posted.

These research products have been installed on an internal server at RDECOM and they have been demonstrated there with a Xybernaut wearable computer.

What is meant by 'simulation'. In this project, it is expected that many of the architectural and service issues raised could be applied to virtually any simulation that can be used for training. This would presumably include part-task simulators, physical simulators, distributed mission training simulators, and so on. If there is to be a computer-based tutorial control system in any such simulation training system, it will need some services from the simulation that are included in the service set specified by the architecture used in this project. As far as the actual iRides *implementation* goes, however, the current iRides system cannot be used for every training mission. The present iRides implementation is particularly appropriate for teaching about systems operation and maintenance. It has been used to prototype software interfaces, but additional work on simulation libraries would be required to make this use of iRides very practical. It cannot be used to build interactive realistic natural views of individual combatant mission trainers. However, it is expected that it could play a role in teaching conceptual (as opposed to perceptual) skills about tactical decision making.

In its current implementation, all iRides simulations make use of a 2D graphical object viewer. Exploratory work (under ONR sponsorship) is being conducted on the feasibility of delivering simple 3D-over-the-web simulation training using the iRides simulation and tutorial engines.

1.3 ADL and iRides.

iRides is a set of software components that deliver interactive instruction and both guided and unguided practice in the context of behaviorally realistic simulations. The concept behind iRides is that it should be possible to distribute over the Internet training systems that closely integrate simulation and advanced pedagogy. The approach is to develop a generic 'player' of simulations and simulation-centered training specifications. Students are able to choose (or can be directed to) a training experience that automatically downloads both the 'player' software and the simulation and instruction specifications that govern the training experience. See Figure 2.

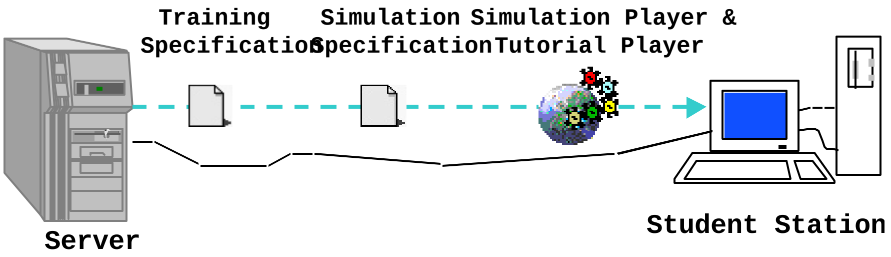


Figure 1. Internet Delivery of Specification-driven Training

During 2002, iRides has been tested in a new delivery mode with potential immediate impact to the warfighter. Figure 2 shows a configuration in which a server with a wireless network card is used to provide simulation data, instructional specifications, and the iRides delivery software to a soldier equipped with a wearable computer that has a compatible wireless network interface.

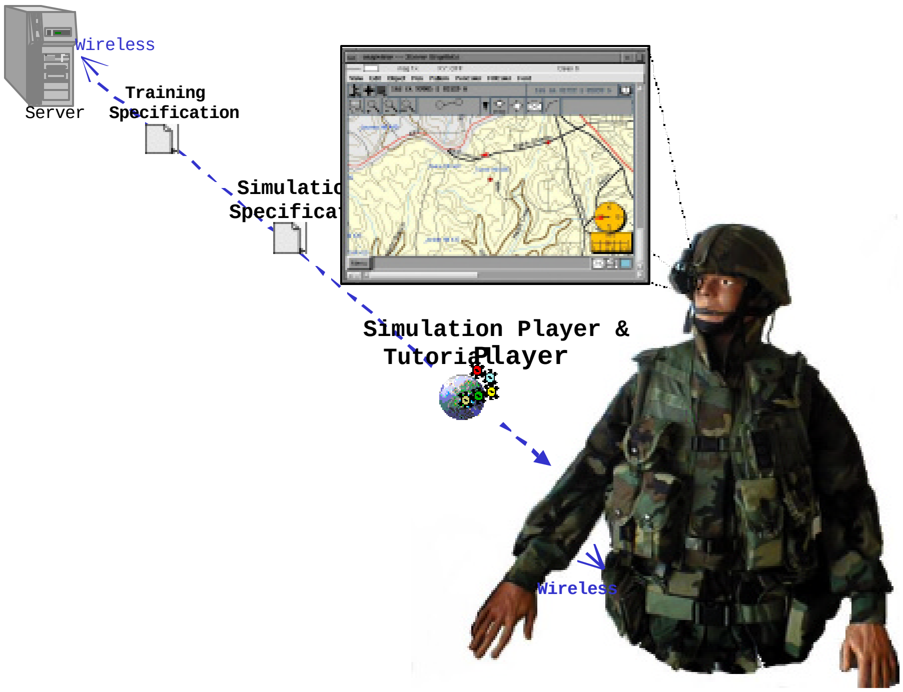
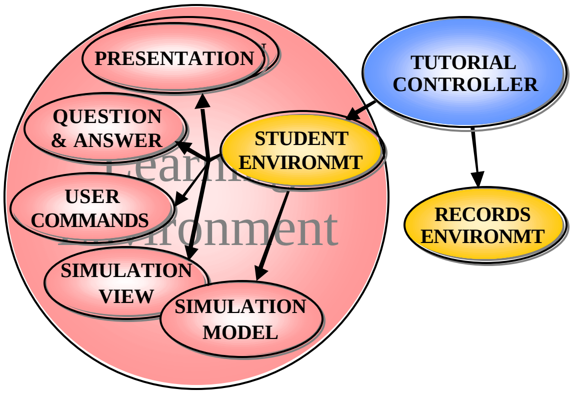


Figure 2. Wireless Delivery of Specification-driven Training

1.4 iRides Architecture and Implementation.

iRides is an implementation of an open architecture for distributed simulation-centered training systems. This architecture prescribes a set of components, shown in simplified form in Figure 3. Figure 4 presents a view of one possible implementation of the architecture, using a 3D simulation view.



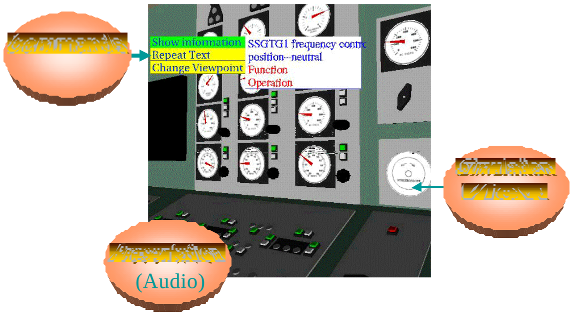
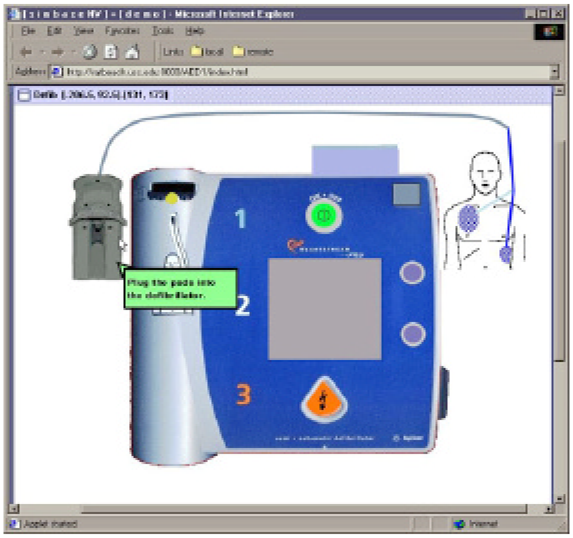
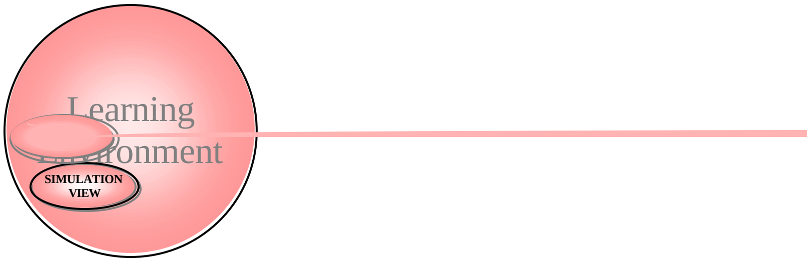


Figure 3. iRides Architecture *Figure 4. A Possible Implementation*

In this architecture, a *student environment* is managed by a tutorial object that determines the pedagogical events that will take place, that evaluates student actions, and that directs the recording of student data. The details of student data maintenance are handled by a Records Environment object. Figure 5 shows a recent version of iRides, running in Microsoft's Internet Explorer, delivering an interactive simulation and lesson on the Phillips HeartStream FR2 Defibrillator.


Microsoft Internet Explorer
File Edit View Favorites Tools Help
Links local remote
Address
1
2
3
Applet started
Internet

Plug the pads into the defibrillator.



PRESENTATIONPRESENTATIONTUTORIALCONTROLLERTUTORIALCONTROLLERSIMULATIONMODELSIMULATIONMODELSTUDENTENVIRON used to produce interactive graphical simulations for training. The resulting products are not only produced at low cost, but also can be reliably maintained. The "Sim data" files contain both graphical and behavioral prescriptions that are used by two iRides components, a Simulation View and a Simulation Model. The former is responsible for the appearance of simulated objects; the latter is responsible for their behavior. These two components communicate with each other. In effect, the Simulation View hears about changes in the Simulation Model that require redrawing of objects. (For example, a Model might change the color, size, location, or orientation of an object, perhaps as a result of some action taken by the student, or perhaps as a result of the simulated decay of the charge in a battery.) Because of the abstraction of the iRides architecture, it will be practical to replace the 2D simulation view used in iRides with a 3D viewer without having to modify other components, such as the Simulation Model. This means that rule-governed authored simulations can be developed for 3D tutoring without having to develop or utilize a different underlying simulation engine

The student environment object shown schematically in Figure 4 and Figure 6 provides a software interface that the tutorial component can use to access the relevant features of all the other components of the learning context, such as the simulation, text and video presentations, and question answering interfaces. In effect, this environment offers a set of services to the tutorial controller. Some of the simulation-specific services are described here.

1.5 iRides Authoring Systems.

Figure 6 shows that the authoring process creates two different types of data files that are used by iRides. (Of course, there are many other types of data files used by presentation components, such as HTML files, image files, and video and audio files. The two file types under discussion here are unique to the iRides delivery and development systems.) One of these is a simulation specification; the other is a tutorial specification. The tutorial specification is used by the tutorial controller to manage the entire instructional process. The simulation specification determines the appearance and behavior of the simulation. Because every authored simulation in the iRides system is delivered by the same universal simulation delivery software, each such simulation is able to provide services that can be exploited by tutorials, including those described above.

Although the basic concepts of authoring simulations for training and authoring tutorial control specifications are not trivially simple concepts, the goal of the development of iRides Author is to make it significantly more productive to develop and maintain these systems than would be the case using programming languages. Simulation training authoring should be no more difficult than developing an Excel spreadsheet with macros, rather than as difficult as writing an application in Java or C++. One reason that a lower level of development skill is required is that simulation authors need not be concerned with issues of the flow of control during an authored simulation. In most cases, the underlying dependencies among behavior rules can be determined by iRides, so the author can focus on relationships, rather than on flow of control issues.

1.6 History.

The design and development of iRides was based on the lessons of two earlier projects, RIDES and VIVIDS. The RIDES and VIVIDS applications were both complete authoring systems and interactive student delivery systems. They were implemented for Unix systems, and could be run on SCO Unix, Linux, Sun OS and Solaris, and on the SGI IRIX operating system. The original motivation for the iRides project was to make it possible to distribute RIDES/VIVIDS simulations and instruction over the Internet. The plan was to develop a completely new, Java-based 'player' that would deliver interactive simulations and tutorials that had been developed in VIVIDS or RIDES.

Native RIDES/VIVIDS file formats were a complex binary representation that included all the cross references among authored simulation objects and authored instructional events. One of the goals in the iRides project is to provide an open architecture that will make it possible for other researchers and developers to use only certain of the iRides components, while substituting conformant software objects of their own invention to fill other roles. A well-defined open architecture for simulation centered learning systems helps to ensure both the maintainability and the reusability of the iRides components. Because the simulation components might be used independently of the instructional components of iRides, the data that each component made use of (simulation specifications and instructional specifications, respectively) were separated into different files in iRides. The formats of these files are human-readable text formats, rather than the specialized binary representation used in RIDES/VIVIDS. As a result, they can now be efficiently version-controlled, which was heretofore not practical. Therefore, multiple developers can more successfully collaborate on trainer development.

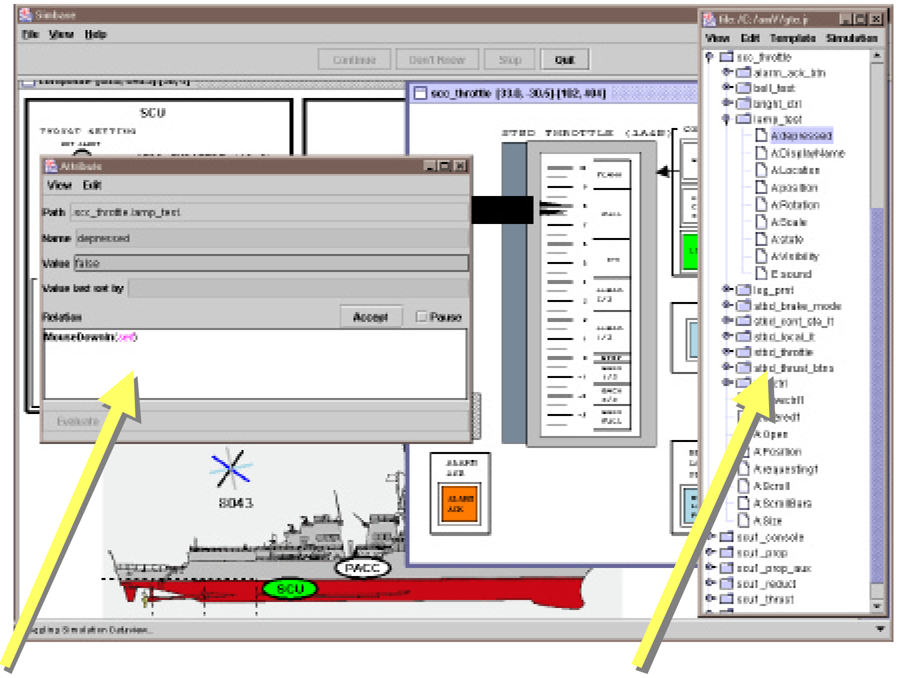
In order that iRides would be able to deliver simulations and instruction previously developed using RIDES and VIVIDS, the last release of the VIVIDS program was given the ability to write out data in the new, textual file formats. Therefore, one way to create web deliverable simulation-centered training materials for iRides is to use VIVIDS to translate an existing RIDES or VIVIDS data file into two files, a simulation file and an instructional specification file, that can be read and interpreted by iRides. Naturally, this version of VIVIDS (often called *classic* VIVIDS) can also be used to author new simulations and training from scratch, and files with the iRides formats for simulation and instruction can be exported. The box labeled "Authoring" in Figure 6 can be seen as classic VIVIDS, which can produce both simulation and instruction specification files.

There are two concerns about relying on classic VIVIDS as the source for all iRides specifications. One is that VIVIDS runs only on Unix or Unix-like platforms, such as Linux. It can be awkward and inefficient to develop on one platform and then test and deliver on another. In addition, computer platforms capable of running iRides are very much more commonly available to instructional developers than are Unix platforms. Any PC with a Windows 95 or later operating system can run Java 1.4 and, therefore, iRides. Clearly, more potential authors could make use of the iRides system if they could author simulations and lessons on any computer that can run Java.

Possibly a more serious concern with relying only on VIVIDS for authoring iRides simulations and instructional specifications is that iRides incorporates many advanced features and options that were not part of VIVIDS. Reliance on only the VIVIDS simulation and instruction features would greatly limit the productive use of iRides.

1.7 Authoring Simulations in iRides.

Work is being carried out to introduce authoring capabilities into iRides (under ONR funding). This will make it possible to develop and test on the same platform. More importantly, authors will be encouraged to take advantage of the enhanced features of iRides in the context of an iRides-native authoring environment. A set of iRides releases code-named Simbase in early 2002 provided simulation authoring facilities in the iRides environment. A second set of iRides releases code-named InSim provided instruction authoring capabilities. Naturally, the InSim releases continue to offer the SimBase simulation authoring features together with additional enhancements.



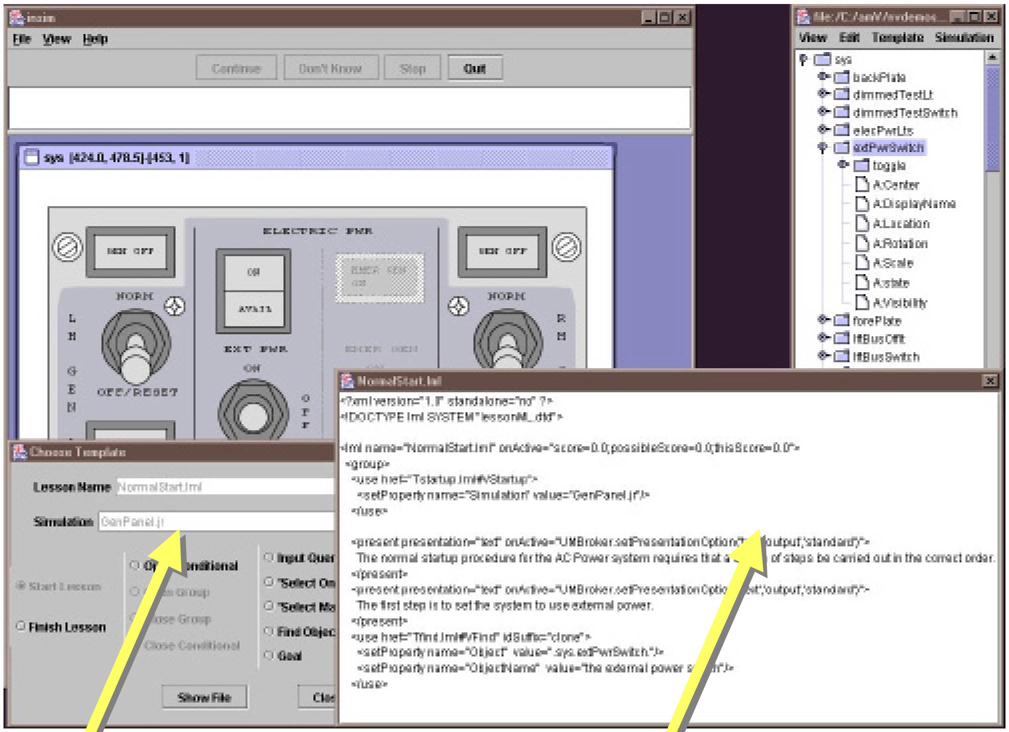
Path scu_throttle.lamp_test
Name depressed
Value false
Value last set by
Relation
MouseDownIn(self)

Attribute View **Simulation Objects List**

Figure 7. Simulation Authoring in iRides

Figure 7 shows iRides being used for simulation authoring. The *simulation objects list* gives the author access to the objects (and sub objects) of the simulation. Authors can also select individual attributes of the simulation. In the *attribute view*, shown at the left in Figure 7, an author can change the value of an attribute or write a rule that will determine the attribute's value. Certain attributes, with names like Scale, Rotation, Visibility, FillColor, FillPattern, etc., directly determine the appearance of objects. Other attributes, introduced by the author modeling the simulated system, may have rules that change values that directly or indirectly affect the values of the graphical attributes. By modifying the values and rules of a simulation, an author determines how it looks and how it reacts to user actions and to the passage of time.

Figure 8 shows the process of authoring a lesson in the iRides instruction authoring system of one of the InSim releases of iRides. Authors work through a series of dialogs that allow them to build elaborate lessons by invoking the use of previously authored instructional components. These reusable instructional specifications are sometimes called templates, but, in fact, any legal Lesson Markup Language (LML) file can be used as a template by another file. The lesson authoring interface includes dialogs that are designed to work with a set of standardized instructional templates, but this system can be easily extended.



Lesson Name NormalStart.lml

Simulation GenPanel.jr

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE lml SYSTEM "lessonML.dtd">

<lml name="NormalStart.lml" onActive="score=0.0;possibleScore=0.0;thisScore=0.0">
 <group>
  <use href="Tstartup.lml#VStartup">
   <setProperty name="Simulation" value="GenPanel.jr"/>
  </use>

  <present presentation="text" onActive="UMBroker.setPresentationOption('text','output','standard')">
   The normal startup procedure for the AC Power system requires that a [illegible] of steps be carried out in the correct order.
  </present>
  <present presentation="text" onActive="UMBroker.setPresentationOption('text','output','standard')">
   The first step is to set the system to use external power.
  </present>
  <use href="Tfind.lml#VFind" idSuffix="clone">
   <setProperty name="Object" value=".sys.extPwrSwitch."/>
   <setProperty name="ObjectName" value="the external power switch"/>
  </use>
```

Template Dialog **Generated LML (XML) Specification**

Figure 8. iRides Instruction Authoring

1.8 The FY 2002 iRides Releases and Their Major Features and Benefits.

The following table (Table 1) presents the major releases of iRides and the features introduced in each release. Many of the features described were developed under funding from the Office of Naval Research, and the benefits of that work have been and will continue to be utilized in the conduct of RDECOM-sponsored projects, including that reported here.

Release	**Release Date**	**Features**	**Benefits**
iRides 0.3 (*Simbase*)	2/21/02	***Simulation Behavior Authoring Release***	
		Simulation behavior authoring in iRides environment	Simulation behavior can be authored or modified without using (Unix-based) classic VIVIDS
		Hierarchical view of simulation objects, events, and attributes	Immediate access to editing environment for any behavior-controlling feature of a simulation
		Attribute data views & event data views	Change any value and any value-determining rule in an authored simulation
		Simulations can schedule attribute value changes with a specified delay	An easier and less resource-intensive way of scheduling simulation effects that are triggered by simulation interactions
iRides 0.4 (*Simbase*)	4/19/02	***Advanced Question-and-Answer Components Release***	
		Generalized architecture for *entries* (question-and-answer objects)	A more flexible and more extensible approach permitting a wider range of question-and-answer assessment actions
		Entry options	A single type of question-and-answer interface can be used in a wider variety of ways, giving instruction authors more flexibility
		Multi-part entries	All the parts of a multi-part question can be answered before the answers are evaluated. Delayed assessment gives students the ability to change answers to earlier parts before being assessed.
		Table entries	Authors can require that a table be filled out as a new form of assessment.
		Optional compressed file format supported	The .irdz file format is used for compressed archives of LML & .jr files. Speeds download times; simplifies distribution and installation of proof-of-concept training modules.
iRides 0.5 (*InSim*)	5/30/02	***Instruction Authoring Release (Alpha)***	
		A sequence of instructional events can be authored by making choices in dialogs in the iRides environment	New lessons or portions of lessons can be authored without using (Unix-based) classic VIVIDS
		Reusable libraries of Javascript functions	Authors can create files of Javascript objects with instructional uses; these files can be reused in a wide variety of lessons.
		New approach to instructional templates	Libraries of re-usable instructional techniques can now be created and used effectively. Properties and attributes of the instructional items in these templates (LML files) can be re-assigned for each use of the template. This makes the iRides instruction system extendable, open-ended.
		LML parsing improvements (SAX parser)	A faster and more modern XML parser is the foundation of the LML interpreter .
iRides 0.52 (*InSim*) (&	6/28/02	***Instruction Authoring Release (Beta)***	

classic VIVIDS 0.99)			
		A standard set of basic instruction templates	Core reusable instructional elements ('Find the object', 'Respond to a particular simulation state', 'Require that the student perform an action', etc.) can now be authored with many fewer lines of LML text. Much more readable, more maintainable lesson specifications result.
		VIVIDS refined to output lessons using the standard set of templates.	Much smaller, more readable, more maintainable LML versions of classic VIVIDS lessons. Faster parsing of lesson specifications.
		Simulation can track mouse position even when mouse is not depressed	Extends the range of user interactions possible in simulations. Authors can turn this feature on and off to optimize performance.
		Improved error handling in the instructional system	An improved run-time environment for identifying problems or bugs
		Integration of CRESST Knowledge Mapper	Advanced assessment methodology can be employed within iRides lessons
iRides 0.6 (*InSim*)	7/26/02	***Enhanced Simulation and Improved Instruction Release***	
		Transparent graphics	Possible to simulate semi-transparent objects
		Graphical *textures* as fill patterns	Improved realism for drawn objects
		Graphic lines definable in terms of end points	Simpler rules can be used to position lines that stretch between moveable points in a simulation
		Rules can catch more types of mouse events	Double-clicking and right mouse actions can now be discriminated and responded to in simulation rules. This provides additional flexibility and power to the author.
		Tutorial can demonstrate actions in a simulation—Playing user actions	Instruction can demonstrate a procedure by 'playing back' a set of low-level mouse & key actions that comprise the procedure.
iRides 0.65 (*insim*)	8/10/02	***WebStart Release***	
		WebStart Supported	Web delivery can be accomplished either with an applet in a browser or using Java WebStart, which results in faster startup times when any iRides course is used for the second or later time
		View LML (XML) text of lesson during authoring	Authors can see exactly what has been specified so far in a lesson. Reduces authoring errors.

Table 1. The iRides Releases of FY 2002

Notes on Table 1.

Release 0.4 expanded the set of assessment options from menu choices and numeric keypad entries to these:

Fixed Choices, Single Answer possible

Radio button interface

Menu

Fixed Choices, Multiple Answers possible

Check box interface

Menu (with *Close/Accept*)

Numeric keypad answers
Free text entry
Table of answers

Within a table of answers, given cells within that table can be any one of the following types:

free response text
free response numbers
a popup menu of answer choices
a popup radio button group
a popup check box group

Students can fill out the cells of a table assessment interface in any order, touching a cell to activate it. When the student is ready to be assessed, he or she submits the table (by clicking on a *submit* button) for evaluation.

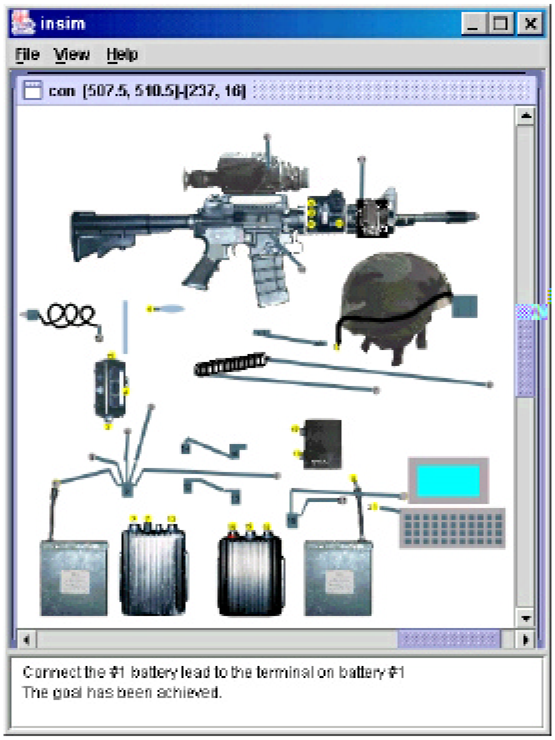
2.0 Research Methodology and Results

2.1 Overview

2.1.1 Three authored proof-of-concept training modules.

Three very different simulation proof-of-concept training modules were developed under the OIST research to conduct and demonstrate the research tasks and objectives described in Section 1.1 (Page2). They are (1) an automatic external defibrillator operations proof-of-concept training module, which is pictured in Figure 5; (2) a Land Warrior connections proof-of-concept training module, shown at the right in Figure 9; and (3) a Land Warrior GPS 2D simulation, which is partially shown as the environment being viewed through a head mounted display in Figure 2.

In each proof-of-concept training module, an interactive simulation accurately portrays the behavior of interest. Instructional vignettes observe student interactions, offer corrections or assistance, and demonstrate how tasks are carried out.

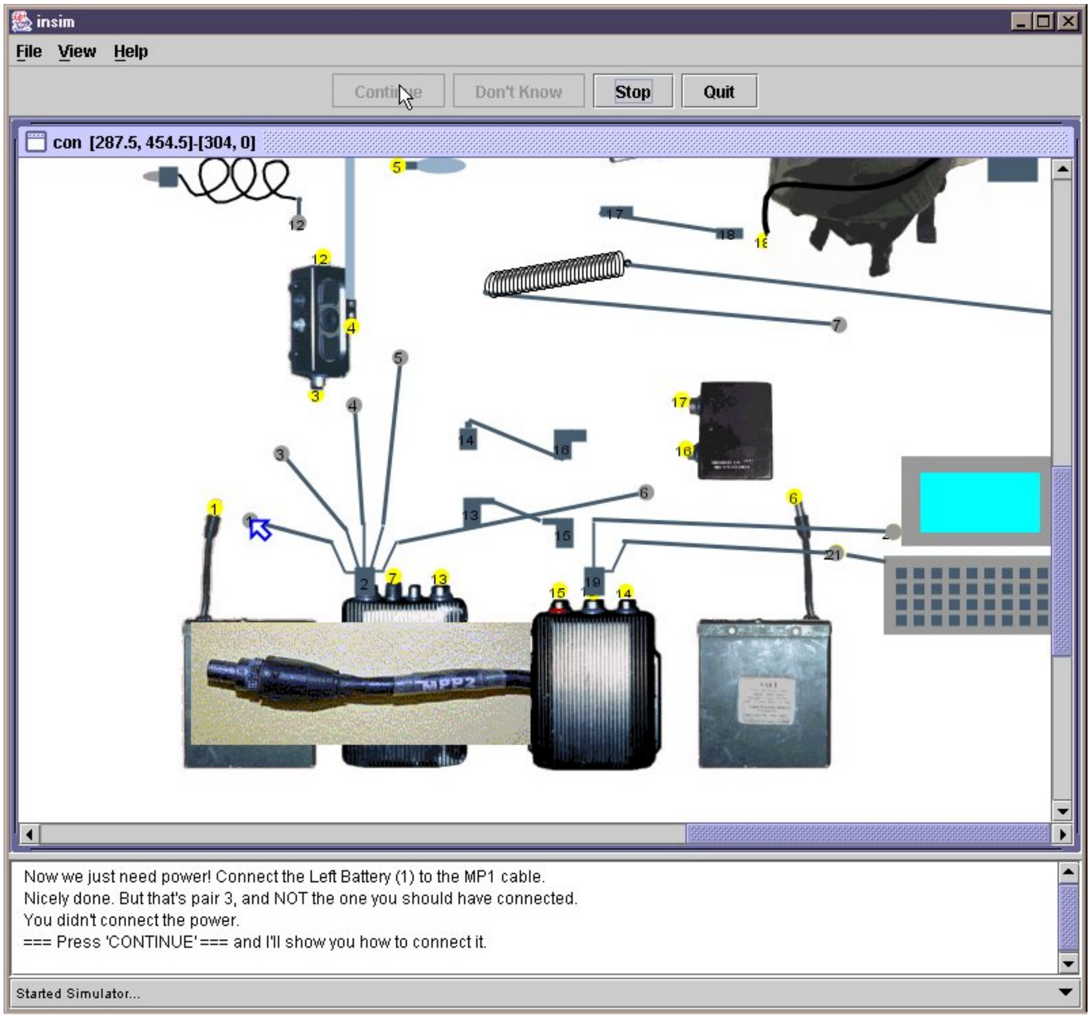


Connect the #1 battery lead to the terminal on battery #1
The goal has been achieved.

Figure 9. Land Warrior Components Proof-of-concept training module

Unlike conventional lock-step computer based instruction (CBT), the tutorial controller is able to observe student actions in a free-play simulation environment and to make instructional interventions as appropriate. Proof-of-concept training modules developed with iRides have the ability to intervene in a simulation environment to deliver contextually appropriate instruction as needed. See Figure 10, below.

Whenever the learner holds the mouse button down while pointing to a cable connector, a larger-scale picture of the connector pops up in the lower part of the display. This feature is designed to help the soldier learn what each cable connector looks like. Similarly, when the virtual instructor 'holds down the mouse button' on a cable connector, the same feature is exhibited, as in Figure 10.



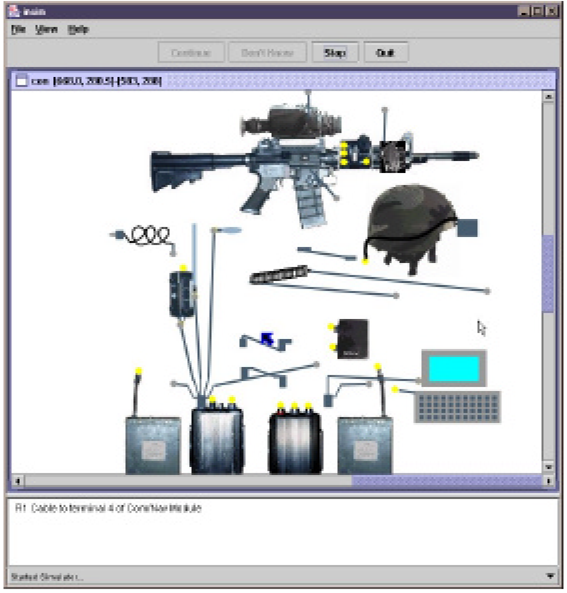
Now we just need power! Connect the Left Battery (1) to the MP1 cable.
Nicely done. But that's pair 3, and NOT the one you should have connected.
You didn't connect the power.
=== Press 'CONTINUE' === and I'll show you how to connect it.

Started Simulator...

Figure 10. Detecting and Remediating a Learner Error

iRides instruction authors can illustrate what should be done in the context of a simulation by directing that a virtual mouse pointer move in the scene. This virtual mouse can also click or press and hold, and it can drag objects. Any simulation effects that would result from the learner taking one of these actions will also occur as a result of the iRides virtual instructor taking the action.

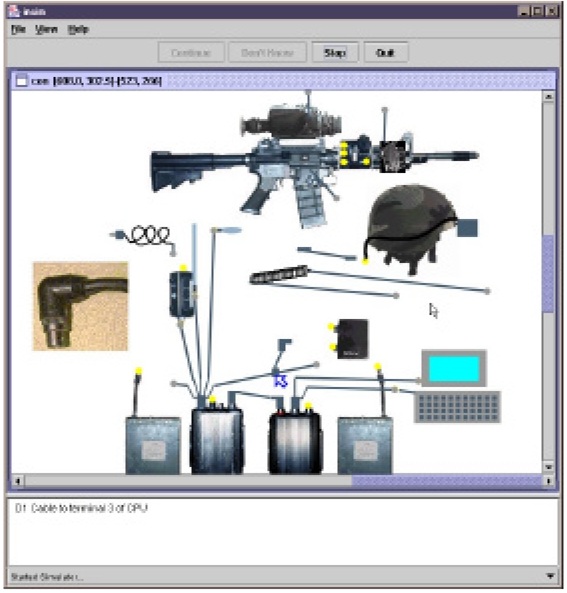
As can be seen in Figure 11, when the virtual instructor's mouse pointer is shown, it is a solid blue color (just to the left and below the center of the scene). When the virtual mouse button is pressed as part of a demonstration, the pointer is shown in blue outline with a white fill area, as in Figure 12 (where the pointer appears as an open pointer in a black and white version of the image). This virtual mouse pointer improves the original demonstration feature of iRides, which did not have a visual element such as the moving mouse pointer to make clearer what the virtual instructor was doing.



R1 Cable to terminal 4 of CommNav Module

Started Simulation...

Figure 11. Virtual Mouse Illustrates Required Move



D1 Cable to terminal 3 of CPU

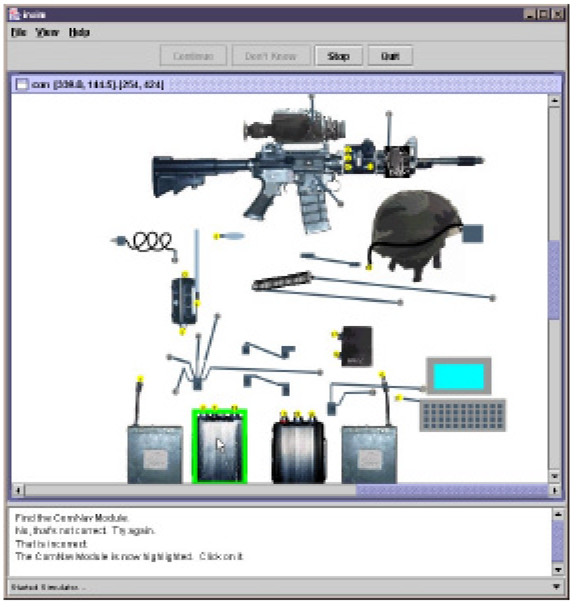
Started Simulation...

Figure 12. Virtual Mouse Carries Out Required Drag

The OIST simulation-centered lessons exhibit several advanced features for distributed learning applications.

- Scripted demonstrations in the context of simulations
- Detection of simulation states that can prompt instruction (opportunistic instruction)
- Highlighting in the simulation under the control of the instructional script (Figure 13)

One lesson segment introduces the names of the land warrior objects and then drills the student on those names. Students must repeat the drill until a criterion number of correct responses are made during a drill. The drill works by presenting the name of the object and requiring that the learner click the mouse on that object. Two chances are given to point out the correct object, and only partial credit is given for a correct second attempt. If the user fails to identify the named object on the second attempt, a remediation using highlighting is presented, as shown in Figure 13.



Find the CommNav Module.
No, that's not correct. Try again.
That is incorrect.
The CommNav Module is now highlighted. Click on it.

Started Simulation...

Figure 13. Remediating an Object Misidentification

2.2 Objectives, Evaluation Criteria, and Results

Four objectives and specific evaluation criteria were identified for this project.

2.2.1 Objective 1:

Demonstrate interactive 2D simulation training using a wearable computer system with head-mounted display.

Evaluation Criterion:

Successfully demonstrate 2D simulation training using a Xybernaut MA-V Wearable Computer with a MicroOptical CO-3 VGA Head Mounted Display.

Results:

Two Land Warrior simulation proof-of-concept training modules were developed using iRides to fit in the 640-by-480 display of the MicroOptical head mounted display. Fig14 shows a view of the LW Connections training environment. (A version that can make use of additional screen real estate has also been developed. See, for example Figure 10.)

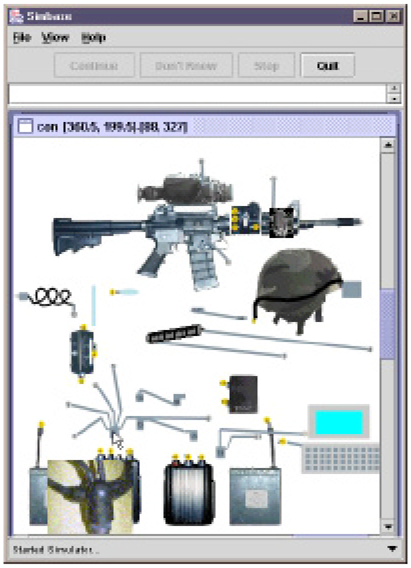


Figure 14. LW Connections Proof-of-concept training module

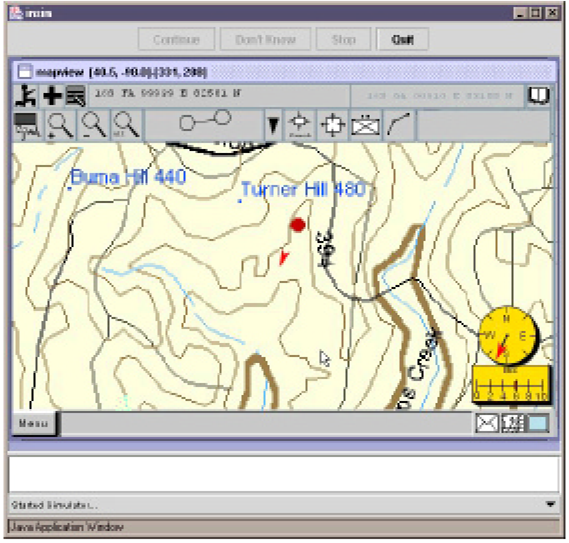


Figure 15. LW GPS Proof-of-concept training module

Fig15 shows the GPS system simulation in action. Both of these applications have been demonstrated on the Xybernaut system. They can also be demonstrated on a wearable computer at the RDECOM laboratory .

2.2.2 Objective 2:

Demonstrate on-demand distribution of 2-D simulation training by delivering *over a network* in the context of an HTML browser.

Evaluation Criterion:

Successfully demonstrate interactive 2D simulation training with all the characteristics described in the Evaluation criterion for Objective 1. *In addition*, this training material must be provided, not by an application already present on the wearable computer's storage systems, but rather by code downloaded on demand that runs in a browser window. It should be possible to click on a link in a browser window and have the interactive training environment downloaded over an 822.11(b)-compatible wireless network. The effect on load time of using Sun WebStart to access content via a network will also be demonstrated. This will be particularly important if a full applet download proves to be alarmingly slow.

Results:

This objective was achieved in two different ways. The iRides delivery system was developed both as an applet and as a Sun WebStart application. In either case, the student clicks on a link in a browser to request that a server provide the iRides application and the simulation and training specifications. In the case of the applet version, a frame within a browser page serves as the container for the iRides delivery system. (See Figure 16.) In the WebStart version, a separate window opens to provide iRides and its training content. (See Figure 17.)

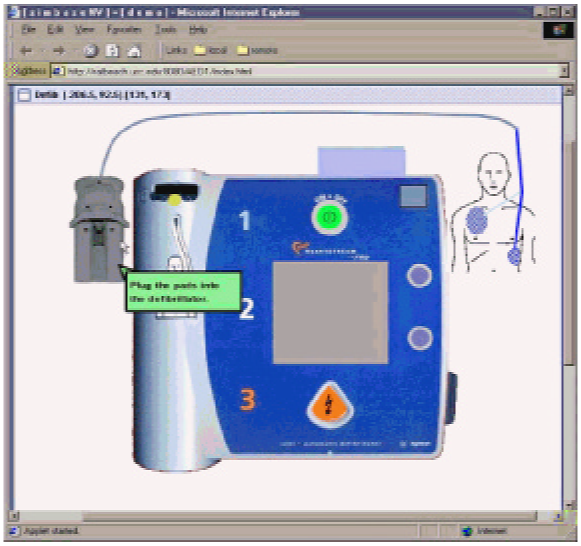

File Edit View Favorites Tools Help
Links
local
Address
Defib
Plug the pads into the defibrillator.
1
2
3
Applet started.
Internet

Figure 16. iRides Applet in Browser

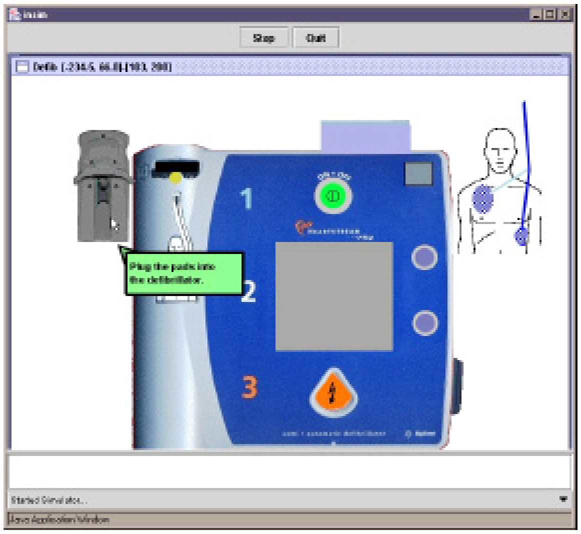


Figure 17. iRides WebStart Application

This research contract provided the support that made possible the development of a WebStart version of iRides. The WebStart iRides was found to provide noticeably faster downloads on repeat visits. It also provided reassuring indications of download progress that were absent during the process of applet downloads. The WebStart version has proven so successful, that it has now become the standard modality for distributed learning with iRides. Of course, it is also possible to use the application version of iRides, if desired, for delivering authored training without making use of a network or of a browser.

2.2.3 Objective 3:

Demonstrate three distinct modes of interactive instruction in iRides training environments.

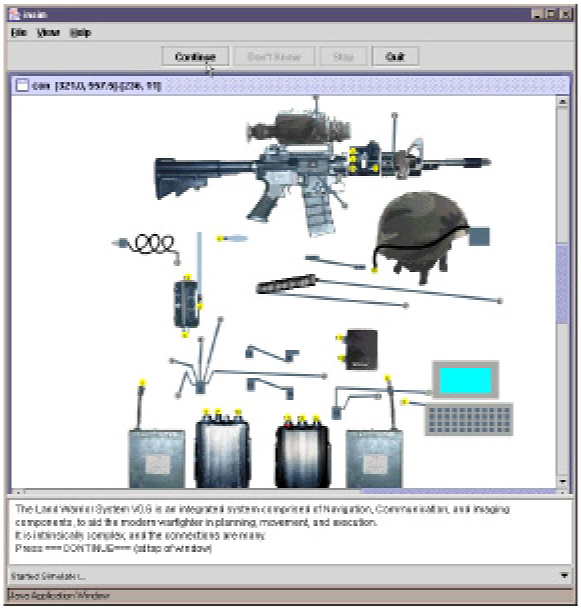
Evaluation Criterion:

Three modes of instruction to be demonstrated are:

- *Introduction* to a task. Explanation and guidance should lead the warfighter through the required steps of the task.
- *Guided practice*. The soldier should be coached through the correct performance of the task.
- *Unguided practice* with opportunistic instruction. The soldier should be able to interact with a 2D simulation with behavioral fidelity. When a soldier makes an error that can be used as an opportunity for instruction, that instruction should be presented.

Results:

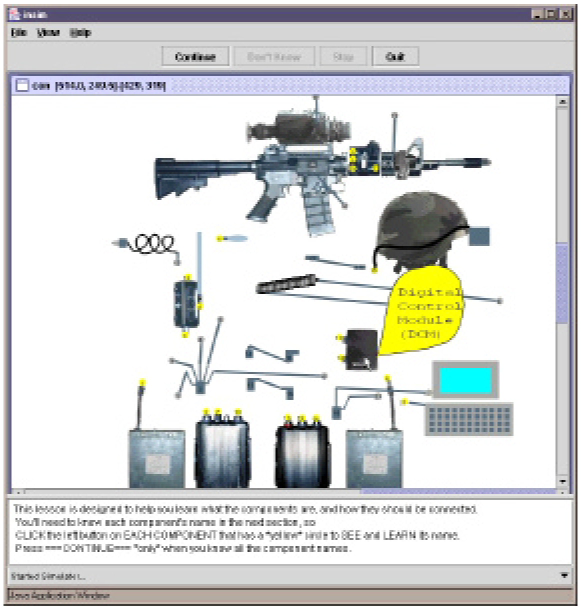
The attainment of this goal is illustrated in the Land Warrior (LW) connection lesson that is illustrated in the following eight figures. This simulation can be accessed from this URL: http://btl.usc.edu/WS/.



The Land Warrior System V0.6 is an integrated system comprised of Navigation, Communication, and Imaging components, to aid the modern warfighter in planning, movement, and execution.
It is intrinsically complex, and the connections are many.
Press ===CONTINUE=== (at top of window)

Figure 18. Introduction to LW Connections

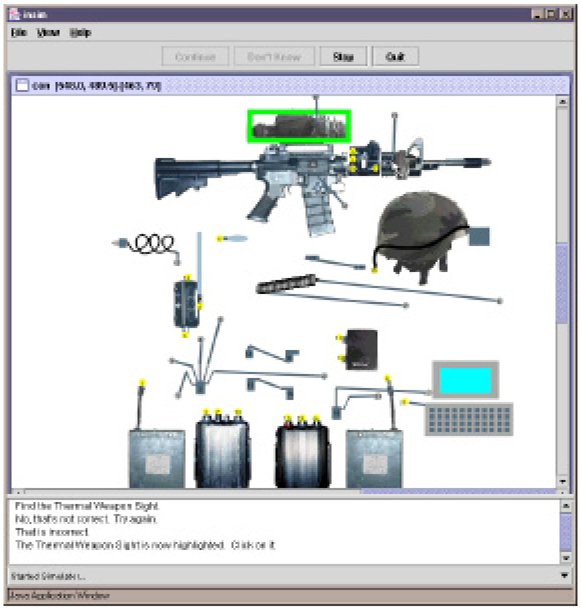
The lesson begins with a brief textual introduction to the task, explaining that the LW connections are complex and must be learned.



This lesson is designed to help you learn what the components are, and how they should be connected.
You'll need to know each component's name in the next section, so
CLICK the left button on EACH COMPONENT that has a "yellow" circle to SEE and LEARN its name.
Press ===CONTINUE=== "only" when you know all the component names.

Figure 19. Learn Component Names

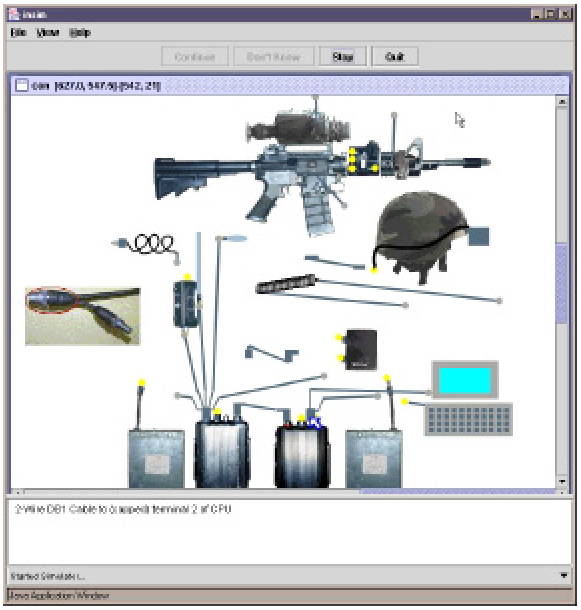
The soldier is then presented with an environment that supports practice in learning the names of the components of the LW system. When the soldier points to an object and holds down the left mouse button, a balloon with a label that names the component appears. The student practices with this environment until he is ready to be evaluated.



Find the Thermal Weapon Sight.
No, that's not correct. Try again.
That is incorrect.
The Thermal Weapon Sight is now highlighted. Click on it.

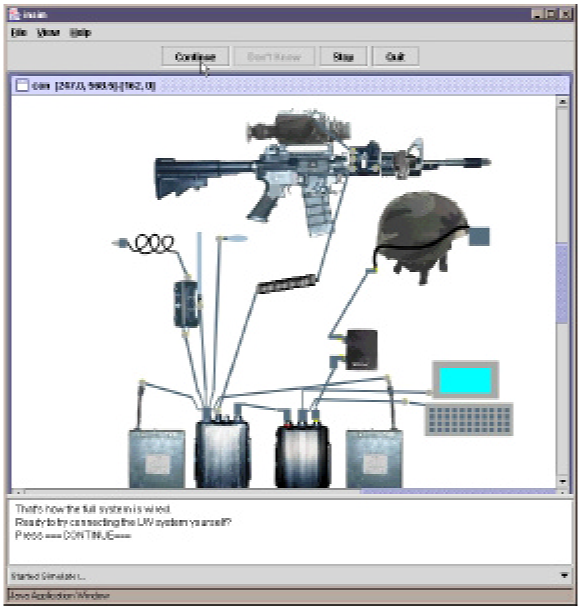
Figure 20. Remediation after Errors

The soldier's knowledge of the component names is then tested. Names are presented randomly, and the soldier is to point to the named object. After repeated errors at identifying a named object, the tutorial provides remediation, and that identification is scored 'incorrect'. If the student does not meet a high identification standard, he repeats the identification exercise until his score does meet or exceed the criterion.



2-Wire DB1 Cable to clapped terminal 2 of CPU

Figure 21. Tutor Demonstrates an Action

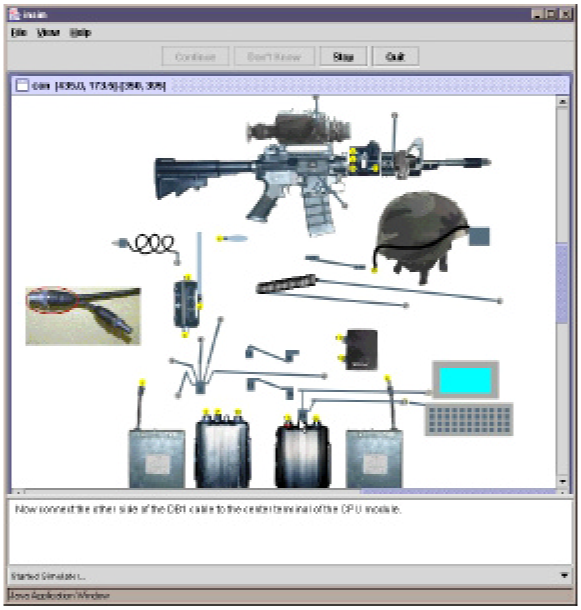


That's how the full system is wired.
Ready to try connecting the LW system yourself?
Press ===CONTINUE===

Figure 22. Invitation to Practice

The next stage of the LW Connections training begins by explaining the color coding system used for cable connectors in the simulation. Connectors with yellow dots are the stationary, 'target' connectors. Gray dots are found on connectors that can be moved. Clicking on a movable connector makes it possible to move that end of a cable to a new position. (It is not necessary to keep holding the mouse button down.) Another mouse click drops off the connector.

The lesson then demonstrates the actions that are required to connect all the LW components. A virtual mouse pointer with a bright blue outline shows the tutor's moves during the demonstration.
Once the connection demonstration has ended, the soldier is invited to carry out a set of connections.



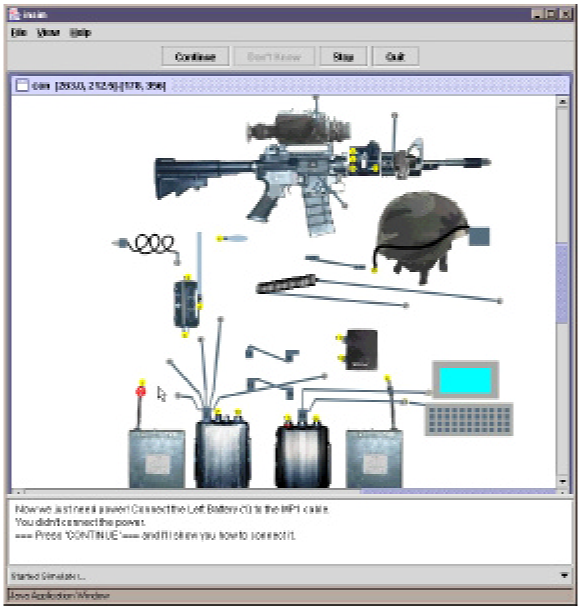
Now connect the other side of the CB1 cable to the center terminal of the CPU module.

Started Simulator...

Java Application Window

Figure 23. Guided Practice

In a guided practice mode, the lesson then directs the soldier through a series of steps that will create a minimal boot configuration of the LW system.



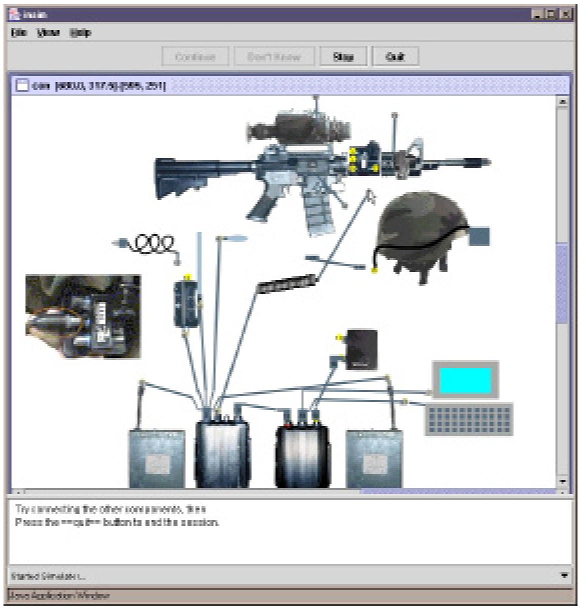
Now we just need power! Connect the Left Battery (1) to the MP1 cable.
You didn't connect the power.
=== Press 'CONTINUE' === and I'll show you how to connect it.

Started Simulator...

Java Application Window

Figure 24. Catching and Remediating Errors in Guided Practice

If a student is unable to correctly carry out a directed step, the lesson shows the student how to carry out that step.



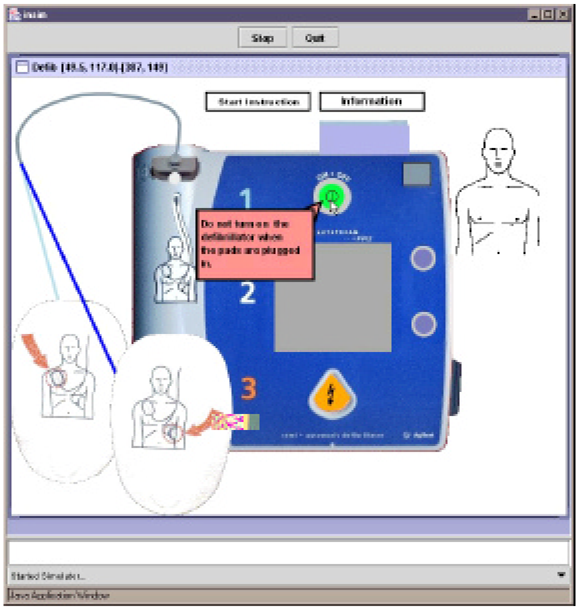
Try connecting the other components, then
Press the ==quit== button to end the session.

Started Simulation...

Figure 25. Unguided Practice

The soldier can continue to connect the other LW components.

Similarly, a variety of modes of operation can be observed in the defibrillator tutor. If a student performs certain inappropriate actions, the training system will detect those actions and provide informative remediation. Here, the student has plugged in the defibrillator pads before turning on the unit. When the student presses the 'ON-OFF" button, the anomalous condition is caught, and the student is informed.



Do not turn on the defibrillator when the pads are plugged in.

Started Simulation...

Figure 26. Opportunistic Instruction

2.2.4 Objective 4:

Demonstrate interactive behavior authoring features.

Evaluation Criterion:

Using the desktop/application version of the OIST software, demonstrate interactive modifications of the relational rules and events that control the graphical simulation. It should be possible to modify any rule or event, using any legal construct of the iRides simulation language, to create a desirable variant of the original simulation.

Results:

Three major authoring interfaces have been developed for interactively editing the behavior of an authored iRides simulation. The author accesses the first of these by using the Simulation Data View command on the View menu. (Only authors have access to the menus shown in Figure 27, at the right. The menu titles do not appear in the student environment.)

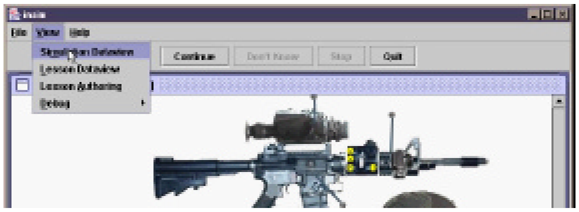


Figure 27. Opening the Simulation Data View

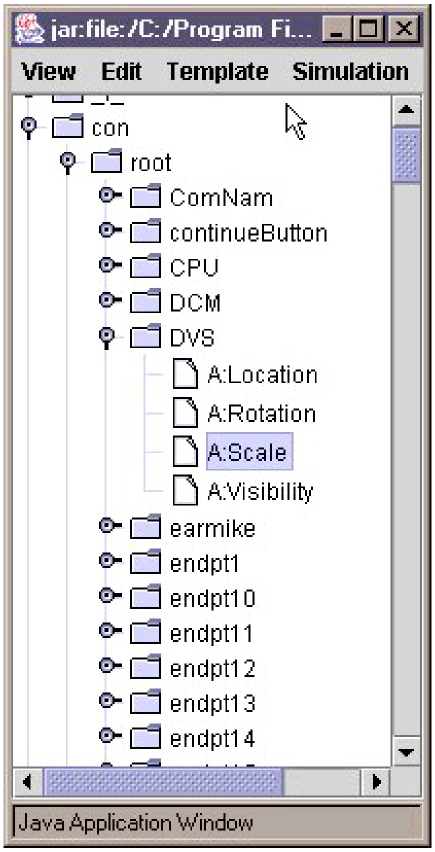
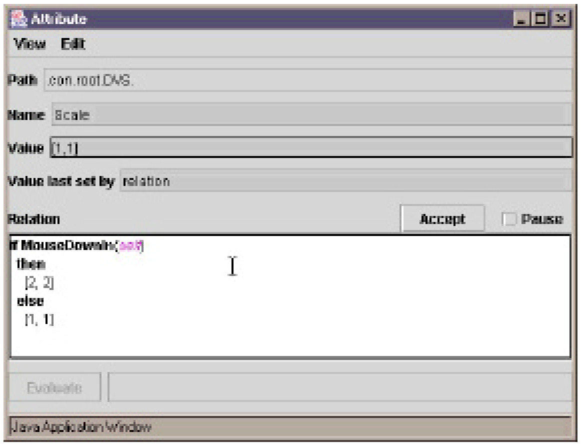


Figure 28. The Simulation Data View

When this command is issued, the simulation data view, shown at the right, appears. Grouped objects, including scenes, can be expanded in this view by clicking on the outline symbols, ⊙-. In the view, the author can see that there is a scene named "con" that has a top level grouped object named "root". This object, in turn, has component objects, including the "CPU"—central processing unit, the "DCM"—digital control module, and the "DVS"—daylight video sight, etc. In Figure 28, the author has also expanded the DVS object, which has four attributes: Location, Rotation, Scale and Visibility. The values of these attributes determine where the DVS is located, whether it appears at an angle or with its original orientation, whether it is at its original size or appears scaled up or scaled down, and whether the DVS is actually visible or not. The behavior of objects in a simulation can be altered by adding or modifying behavior rules that are associated with the their attributes. Double-clicking on the Scale attribute of the DVS object will open an Attribute Data View for that attribute, as shown in Figure 29. If the author wanted this object to appear twice as large as normal when a student points to it and holds down the mouse button, this can be accomplished by entering the rule "If MouseDownIn(self) then [2,2] else [1,1]" as the rule for the Scale attribute. This means that if the mouse button is held down while the mouse is pointing to the object, then it should be both twice as wide and twice as tall as its original size. As soon as this rule is entered and accepted, the editor reformats the rule automatically as shown in Figure 29.



Path .con.root.DVS.

Name Scale

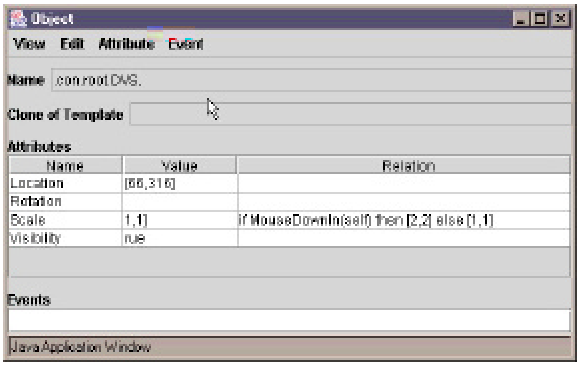
Value [1,1]

Value last set by relation

Relation

```
If MouseDownIn(self)
 then
  [2, 2]
 else
  [1, 1]
```

Figure 29. New Rule in an Attribute Data View



Name .con.root.DVS.

Clone of Template

Attributes

Name	Value	Relation
Location	[66,316]	
Rotation		
Scale	1,1]	if MouseDownIn(self) then [2,2] else [1,1]
Visibility	rue	

Events

Figure 30. Object Data View of DVS

An author can also open a data view of a chosen object. For example, the data view of the DVS object is shown in Figure 30. In either an attribute data view or an object data view an author can change the value of an attribute, and he or she can add, delete, or modify a relation rule that prescribes the behavior of that attribute in the simulation. In the object data view, authors can also add and delete attributes, and they can create new events (behavior elements with multiple steps), delete events, and modify the events themselves.

3.0 Discussion and Recommendations

The above results have shown that it is possible to author interactive graphical simulations that can deliver individualized instruction that works closely with the simulation. Furthermore, these training simulations can be delivered on wearable computers with head-mounted displays, such as the ones being used with Land Warrior.

Simulation alone does not constitute training. Military training simulators are ordinarily used under the supervision of operators and instructors who provide guidance, advice, and after-action reviews. This approach will not work for distance learning, where instructors may not be available at the student's site. Instead, a software component must be used to control and to monitor the simulation for instructional purposes. The iRides delivery infrastructure supports interactive monitoring, evaluation, and control in the context of web-delivered simulation-based learning. This technology supports remediation in the simulation context; natural, interactive task training; and performance assessment in simulated work contexts.

3.1 Lessons Learned

Several broad lessons for warfighter training have been learned as a result of this project and several more specific technical lessons have also emerged from the work. The broad lessons include these:

1. 2D-simulation-centered training can be delivered by lightweight, wearable computers, similar to those of the Land Warrior systems.
2. It is practical to deliver such simulation-centered training over a network from the Internet or an intranet.

3 These compact, downloadable training systems can deliver training in a number of modes, including introductory/demonstration-based, guided practice, and unguided practice.

4 Simulation behaviors for such training can be interactively authored in an environment that is very similar to the learner's runtime environment.

More specific technical lessons learned from this work include these:

5 The pointing devices used in place of a conventional desktop mouse in wearable computer systems can make it difficult to carry out a conventional 'drag' action (moving the pointer while holding down a button). Simulations can be designed to support simpler drag actions in order to reduce the difficulty of using them for soldiers with wearable systems.

6 Applet downloads can be slow for substantial training systems. This problem can be satisfactorily addressed by implementing the training and simulation delivery systems as a WebStart application.

3.2 Future Research Plans

The next challenges for on-demand interactive simulation-centered training are to extend the range of soldier training applications based on iRides in order to address a range of significantly more complex military training applications. For example, a robust approach to tactical training in simulation contexts would benefit from a pedagogical agent that has a flexible, intelligent grasp of salient features of the subject matter. This would make it possible for the training system to recognize tactical errors, for example, not by their specifics, but by their general characteristics. The deeper the pedagogical agent's understanding of the abstract criteria for tactical decision-making, the more practical it will be to provide high quality interactive tactical training at reasonable cost.

Many aspects of small unit planning and tactics can only be learned through field training. Other aspects, however, can be learned through a part task training system. Many of the

- Voice I/O capabilities (2003)
- Advanced graphics capabilities (2003)
- Improved student records/data foundations (2003)
- Automatic collection of performance records (2003-2004)
- Extend core instruction features (action monitoring) (2003)
- Efficiencies, performance tuning (speed, memory footprint improvements) (2004)

Authoring Products for Simulation and Simulation-centered Instruction

- Integrated simulation and instruction authoring (2003)
- Simulation debugging (2003)
- Instruction debugging (2003-2004)
- Integration of COTS (CorelDraw, Illustrator) graphics authoring and simulation authoring (2004)
- Instruction templates and template interfaces (2003-2004)
- Authoring with instructional strategies (2003-2004)
- Documentation for instructors and administrators (2004)
- Documentation for simulation authors and instruction authors (2004)

The following table briefly describes the planned characteristics of the iRides releases planned for FY 2003.

Release	**Release Date**	**Features**	**Benefits**
iRides 0.7	31 Oct 02	***Voice I/O Release***	
		Support for text to speech presentations	Permits instructional information to be delivered by voice, preserving available screen area for simulation content. Will require IBM ViaVoice. Especially useful for training tasks that have high visual attending/tracking requirements.
		Support for voice input: commands and entries	Makes it possible to give students control over instructional features such as requests for help and answering simple assessment questions.
IRides 0.75	29 Nov 02	***Advanced Records Release***	

		Specific knowledge assessments from knowledge mapper	Instruction controller can ask a student's knowledge map about the student's grasp of a particular concept. The specific assessment can then be used to determine which instructional experience should next be provided for this student
		Simulation break points	Authors can specify rules and event statements at which simulation should be halted for debugging
iRides 0.9	7 Months	***Advanced Graphic Simulations Release***	
		SVG graphics Model View	Use of commercial drawing tools to build SVG 2D simulations
		Mid-level instructional strategies	An expanded library of instruction tactics/ strategies for a variety of learning requirements and content characteristics. Improves the quality of instruction while increasing authoring productivity.
IRides 1.0	9 Months	***Complete iRides Authoring Release***	
		Complete non-VIVIDS authoring	iRides proof-of-concept training modules can be authored on Windows (and other compliant Java platforms) computers without requiring use of (Unix-based) classic VIVIDS
iRides 1.1	10 Months	***Instructional Strategies Release***	
		High level templates for specifying instructional strategies	Permits authors to specify instruction based on tested teaching strategies. Reduces authoring difficulty while improving quality and consistency of the training product.
iRides 1.2	12 Months	***iRides Authoring Documented Release***	
		Documentation and authoring examples	Helps new authors learn how to create iRides simulations and iRides instruction. More efficient, debugged development and delivery products for ADDL.

Table 2. Planned Year Two Tasks

3.2.2 Conference Presentation

The research work described here will be presented in a poster session at the Army Science Conference, Orlando Florida, December 2-5, 2002 (Munro, Pizzini, Johnson, Walker, Surmon, Dumanoir, and Garrity, 2002).

4.0 References

Anderson, J. R. (1993). Rules of the Mind. Hillsdale, NJ: Erlbaum.

Antal, J.F. (1995) *Infantry Combat: The Rifle Platoon*. Navato, CA: Presidio Press.

Frederiksen, J.R., White, B.Y., Collins, A., and Eggan, G. (1988) Intelligent tutoring systems for electronic troubleshooting. In Psotka, J., Massey L.D., and Mutter, S.A., (Eds.) *Intelligent Tutoring Systems: Lessons Learned*. Hillsdale, New Jersey: Lawrence Erlbaum Associates.

Johnson, W. L., Rickle, J., Stiles, R. and Munro, A. (1998) Integrating Pedagogical Agents into Virtual Environments. *Presence*.

Munro, A. (1994) Authoring interactive graphical models. In T. de Jong, D. M. Towne, and H. Spada (Eds.), *The Use of Computer Models for Explication, Analysis and Experiential Learning*. Springer Verlag.

Munro, A. (1998) Partially Automating Cognitive Analysis in the Context of a Simulation Authoring Environment. In the *Proceedings of the 7th Annual Conference on Computer Generated Forces and Behavioral Representation*.

Munro, A. (1999) An Open Component Architecture for Simulation-Centered Tutors. In the Proceedings of the Workshop on Advanced Training Technologies and Learning Environments. 9-10 March, NASA Langley Research Center, Hampton, VA.

Munro, A., Johnson, M. C., Pizzini, Q. A., Surmon, D. S., Towne, D. M. and Wogulis, J. L, (1997) Authoring Simulation-Centered Tutors with RIDES. *International Journal of Artificial Intelligence in Education*., **8**, 284-316.

Munro, A. and Pizzini, Q. A. (1998) *VIVIDS Reference Manual*, Los Angeles: Behavioral Technology Laboratories, University of Southern California.

Munro, A., Pizzini, Q. A., Johnson, M.C., Walker, J., Surmon, D., Dumanoir, P., Garrity, P. (2002). On-demand interactive simulation-based training (OIST), Abstract in the Proceedings of the Army Science Conference.

Munro, A., Surmon, D., Johnson, M., Pizzini, Q., and Walker, J. (1999) An Open Architecture for Simulation-Centered Tutors. In Lajoie, S. P. and Vivet, M., *Artificial Intelligence in Education: Open Learning Environments: New Computational Technologies to Support Learning, Exploration, and Collaboration*, Amsterdam: IOS Press, 360-367.

Pizzini, Q.A., Munro, A., Wogulis, J.L., and Towne, D.M., (1996). The Cost-Effective Authoring of Procedural Training, in *Architectures and Methods for Designing Cost-Effective and Reusable ITSs* Workshop Proceedings, ITS'96, Montreal, Québec, Canada.

Sleeman, D. H. & Brown, J. S. (1982). *Intelligent Tutoring Systems*. London: Academic Press.

Towne, D. M. & Munro, A. (1991). Simulation-based instruction of technical skills. *Human Factors*, **33**, 325-341.